

J4 Response to U.S. Public Review Comments on COBOL WD 1.4

* indicates a change was made in response to an item (Sometimes the * is before the comment number and sometimes it follows the number.)

These public review comments are included in this document, ordered by X3J4 document number:

96-0384 Comments on ISO/IEC CD 1989, Prog lang COBOL (Silletti/Wallace)

96-0389 Cmts on CD 1989, Prog lang COBOL; IBM Part 2 (Silletti/Wallace)

96-0412 - public review comments from David DeJongh

96-0414 - public review comments from Raymond Obin

96-0428 - public review comments from Lee Hansen

96-0429 - public review comments from Charles Townsend

considered 96-0442 and 96-0441 as part of this comment

96-0430 - public review comments from Ronald Silletti - Part 3

96-0446 - public review comments from Ronald Silletti - Part 4

96-0447 - public review comments from Robert Sandler

96-0458 - public review comments from John Piggott

96-0461 - public review comments from Jeffrey Friedman

96-0473 - public review comments from Don Schricker

96-0474 - OO public review comments from Don Schricker

96-0476 - public review comments from Ronald Silletti - Part 13

96-0477 - public review comments from Ronald Silletti - Part 14

96-0478 - public review comments from Roger Knight

96-0479 - public review comments from Roger Knight

96-0480 - public review comments from Ronald Silletti - Part 11

96-0481 - public review comments from Ronald Silletti - Part 10

96-0483 - public review comments from Ronald Silletti - Part 5

96-0484 - public review comments from Ronald Silletti - Part 9

96-0485 - public review comments from Ronald Silletti - Part 7

96-0486 - public review comments from Ronald Silletti - Part 8

96-0487 - public review comments from Wataru Takagi

96-0488 - public review comments from Ronald Silletti - Part 6

96-0489 - public review comments from Ronald Silletti - Part 12

96-0490 - public review comments from Clark Morris

96-0493 - public review comments from Ronald Silletti - Part 15

96-0500 - Public review comments from Micro Focus (Gilbert/Gamble)

96-0502 - Public review comments from Ronald Silletti - Part 16

96-0506 - Public review comments from Jonathan Beit-Aharon

Received

DEC 0 8 1999

**Director's Office
Group 2700**



sent Nov. 20, '96

96-0384 R. E. Silletti (IBM Corporation)

*1 Length of figurative constant. The draft does not specify the length of a figurative constant when used in a concatenation expression. IBM has submitted proposal X3J4/96-0342 to Technical Committee X3J4 to amend the document.

Response: Accept. The length of a figurative constant in a concatenation expression was specified as one in 97-0218, Length of figurative constants in concatenation expressions, which was approved by J4 at meeting 208.

- Add four methods (I can't think of any pleasing names) similar to the methods of Comparable, all invariant, but which ignore upper/lower case for the purposes of the comparison.
3. I'm a bit disturbed by the absence of any streaming methods on an object for the purposes of externalization and subsequent reconstitution. These protocols might be used on transient (non-named) objects, not just formally persistent objects.)
4. P 587, 16.3.1.1. Why is "OccurrencesOf" not invariant? How does this method modify the collection self? Same comment for List (p 633, 16.3.16.1).
5. P 623, 16.3.14.1. Why is "Intersection" structurally different from all the other operations that produce a changed IdentitySet, for example, Union? Is it just so that the postconditions can be stated neatly?

Suggestion. Change to ...

AnIdentitySet "Intersection" using Another instead of ...

AnIdentitySet "Intersection" invariant using Another returning AnIntersection

6. P 625, 16.3.14.6.1. Change "method-id. is-subset" to "... IsSubset".
7. P 633, 16.3.16.1. Method "Sublist" should be invariant, right? Just like "OccurrencesOf".
8. Both national character handling and internationalization are important aspects that must be addressed. Dependencies, if any, on the environment of the invoking routine need to be specified.

It's not clear how collating sequences are managed, even with respect to non-internationalized environments. Is the collating sequence in the library always the one in effect at the time of invocation, thus requiring the library to be an "internationalized" implementation with respect to collating sequence? How are the different character types handled (alphanumeric and national)?

Received

DEC 0 8 1999

**Director's Office
Groom 2700**

→ 96-0478 - public review comments from Roger Knights

Here are some comments on your draft Cobol-97 standard. They are in four categories: first, Substantive (mostly date-related); then editorial suggestions dealing with: Miscellaneous, your Substantive-Changes-Not-Affecting list, and RW / Validate.

You needn't write me anything but the briefest note about your action on this letter; you have too much other work to do.

A. Substantive:

1. To facilitate Year-2000 conversion-testing, allow the CURRENT-DATE special register to be both modified and tested-to-see-if-modified. (John Piggott suggested the first of these.) I guess the SET statement could be used for the first purpose. For the second, IF would interrogate a new special register that would be either a True/False "indicator" or a backup location like TRUE-CURRENT-DATE or SAVED-CURRENT-DATE or ACTUAL-CURRENT-DATE. And perhaps there should be some standard way to restore CURRENT-DATE to its original status.

Response: Reject.

2. Your FORMAT clause will assist Year-2000 migration by removing an obstacle to the use of compact date forms, such as the Gregorian-INTEGER format used by COBOL's Functions, and the new Packed and Binary formats that various Year-2000 vendors and writers are proposing. The obstacle is that such formats are harder to download to the PC, which has caused some authorities to advise against converting to them. (E.g., See Tick Tick

Tick, Winter '95, page 8, col. 3.) So don't drop it if someone points out any problem areas it may have. Resolve such questions of interpretation in a subsequent Year-2000 task group (see next item).

Response: Accept. The FORMAT clause is retained.

3a. Consider setting up a Task Group to interface with persons and organizations dealing with the Year-2000 Problem, in case there are other changes like the above that could be implemented as Addenda, or at least so that vendor extensions could be coordinated and informally standardized. Most of the work could/should be done on the Internet, with a special extra site for private (inter-member) task group communications. At a minimum, such a task group could consider minor twiddling like the CURRENT-DATE adjustments I proposed in Item 1, above. More substantial enhancements might well be considered, either as a result of user pressure and/or of the opportunity to make a buck by offering a desperately needed service. It may be that these enhancements will come too late to help most users. Still, they will help some, and they will enable those applications that have been deactivated as a result of "triage" to be reactivated, since less work and (especially) no file conversion will be needed to get them running. The enhancements I have in mind include:

Response: Reject. There are already a number of task groups in the industry.

3b. An extensive family of built-in date-manipulation functions. In (84051) RK-WP Date-Handling Functions, I suggested (in addition to suggesting the current date-conversion functions) a day-counting (between dates) function and a date-comparison function, both of which could accept operands in different formats. Those two basic items could be added in an addenda, I trust. Beyond them are more sophisticated routines that really ought to be made available—there's certainly a demand for them. There are now several vendors offering date-routine collections, including holiday tables that work both domestically and/or internationally. (User adjustments to them are possible too.) Perhaps one of these vendors would agree to license his product at a low rate to those implementors who prefer to buy rather than build. (Such inexpensive licenseability is commonly considered when ANSI hardware standards are developed, so I extrapolate that it's OK for software too, at least in an emergency. Alternatively, end-user sites could be asked to contribute their date routines to the TG. E.g., as I wrote in (83030) RK-WP Suggestion Smorgasbord, Item 9: "I believe Boeing has a collection which will handle everything but Mayan.")

The advantages of making these intrinsic functions are efficiency, reliability, simplicity (fewer CALLs and COPYs), inter-shop standardization (hence reduced training costs and merger-time nightmares), etc. As a business-oriented language, inclusion of such business-oriented routines is acceptable—indeed, desirable. (Other business-savvy languages have such a data-type; e.g., Clarion, as I mentioned 'way back when.) Year-2000 conversion software could generate these functions and thereby upgrade users' software libraries in the process. Currently, code often comes back from such a conversion in a downgraded (patched-looking) state. In the Nov. '84 minutes, p. 16, the CCC accepted only a subset of the date manipulation functions I had suggested; the other suggestions were implicitly rejected by consensus and not brought to a vote.

Response: These will be considered as an enhancement for future standard.

3c. A DATE data-attribute (i.e., a new clause) whereby a user can both specify an item as being a date and also indicate its format. This would be helpful to the date-comparison and day-counting functions suggested above; although in the absence of a date attribute, the functions could infer one from the item's PIC & USAGE. It would also assist other date-related functions that might be added, as well as being helpful to the maintenance programmer and to COBOL-analysis software. Also, Year-2000 conversion software could insert this data-attribute on items it decides (in optional consultation with humans) are dates.

In addition, the attribute could (if extensible) integrate the many idiosyncratic date formats that now exist into standard COBOL in a nearly seamless way. E.g., assume the keyword for the attribute is "DATE-<suffix>", where standard terms for the most common "suffixes" (like JULIAN and YRMODA and YEARMODA) are built-in, and where other suffixes could be added on the fly. (Perhaps an interface could be provided that would allow user-written (or ISV-written) routines to decode/encode other formats to dates with particular user-suffixes.)

The CCC voted down (10-7) using a data-attribute to handle the four date conversions it now provides in favor of functions in the Nov. '84 minutes, p.16.

